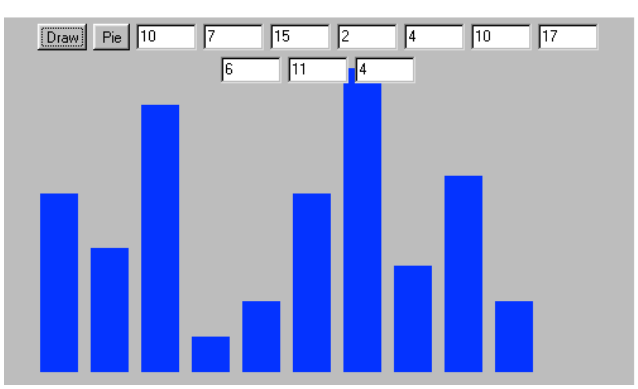
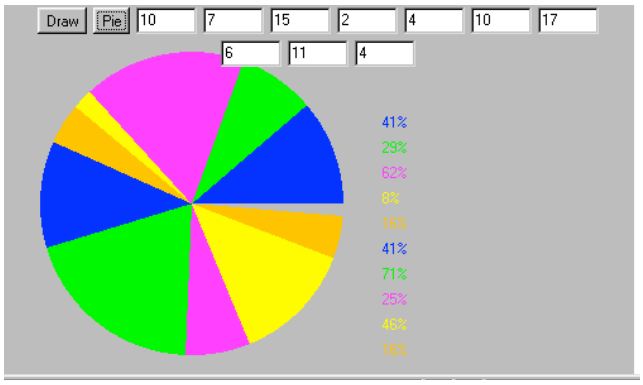
Java – Maths and More Arrays

C&G criteria: 2.1.1, 2.1.2

This exercise takes input from the user and shows it on screen as a bar graph or pie chart. This requires an interface Class and Classes for the 2 types of chart. As the concept of accepting a set of figures and formatting it for output is common to the bar graph and pie chart their respective Classes will share some methods. So 1 will be a sub Class of the other or they will both be sub Classes of some more general super Class. Sets of numbers are best represented with an array so the code will make use of arrays and passing arrays to methods.





This is the code outline for the Bar Class. Note that the data passed around is an array of unspecified length, int[]data. This allows the Class to perform operations on data arrays of varying length. Although the above examples are using arrays with 10 members the Class can handle more or less members. This is useful as the same Bar Class could be used by another interface Class.

```
import java.awt.*;  //needed for Graphics g

public class Bar
{
        public void drawBars(int[] data, Graphics g)
        { // draw the scaled data as filled rectangles using Graphics g
                int largest;
                int []scaleData  = new int[data.length];
                int yStart = 40;
                int yHeight = 240;  //maximum height of bars
                int xStart = 30;
                int width = 30;
                int gap = 10;
                largest = findLargest(data);  //protected method
                scaleData = scaleValues (data,largest,yHeight);  //protected method
                g.setColor(Color.blue);
                int x = xStart;
                for (int count = 0;count<scaleData.length;count++)
                {
```

```
                        g.fillRect(x,yStart + yHeight - //continues on next line
scaleData[count],width,scaleData[count]); //this continues from the line above
                        x = x + width + gap;
                }
        }
        protected int findLargest(int [] array)
        {
                //go through the array of numbers and find the largest
                //record this value as large
                return large;
        }
        protected int [] scaleValues(int []array,int max, int height)
        {//method that takes and sends back an array
                //the largest figure occupies the maximum height allowed by the applet
                //create a new array of the same length as the array[] passed to this method
                //run through each member of the passed array with a for loop
                //scale each member using height/max and assign the result to the new array
                //return the new array – return newArray; – not return newArray[];
        }
}
```

Note the code to pass arrays to and from methods. In no case does the array length have to be specified as the use of array.length allows the code to pick up on the length of whatever length array has been passed.

```
for (int x = 0;x<array.length;x++)
        newArray[x] = oldArray[x] * aVariable;
```

This code will loop through an array adjusting each member in turn providing that the data types of newArray[], oldArray and aVariable are compatible, for instance they could all be integers. For the loop to work it must run from 0 to < array.length as an array of length 5 will have members 0 to 4. Failure to abide by this will throw an array out of bounds error. This is not likely to stop a program running but will stuff the expected result.

Implementation

The sketched out methods need completing and an interface Class creating to accept user input. The data input to the TextFields will be Strings, these need converting to integers for the Bar object. actionPerformed(ActionEvent event) will be the event that draws the graph by calling repaint(). Paint() will then pass the data from the interface to the Bar Class.

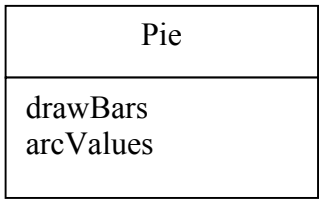
```
public void paint(Graphics g)
{ //in this case figures[] is the array that holds the data derived from the TextFields
        myBar.drawBars(figures,g);
}
```

The Pie chart Class will act in similar manner except that g.FillArc() will be used to draw the data on the applet. Each new arc must start where the previous arc finished off and the total must make a circle. The scaling of values will not be based on the largest member of the array but the fact that the sum of all values when scaled will be 360.

In the example implementation the slices of the pie chart have been given different colours through the use of an array of colours.

```
Color []fill = {Color.blue,Color.green,Color.magenta,Color.yellow,Color.orange};
```

As the slices are drawn the current colour for the Graphics object is set by cycling through the Color array fill[].



In this implementation Pie extends Bar, drawBars has been overridden.